

SSSSSSSSSSSSSS	000000000	RRRRRRRRRRRR	TTTTTTTTTTTTTT	3333333333	222222222
SSSSSSSSSSSSSS	000000000	RRRRRRRRRRRR	TTTTTTTTTTTTTT	3333333333	222222222
SSSSSSSSSSSSSS	000000000	RRRRRRRRRRRR	TTTTTTTTTTTTTT	3333333333	222222222
SSS	000	RRR	TTT	333	222
SSS	000	RRR	TTT	333	222
SSS	000	RRR	TTT	333	222
SSS	000	RRR	TTT	333	222
SSS	000	RRR	TTT	333	222
SSS	000	RRR	TTT	333	222
SSSSSSSSSS	000	RRRRRRRRRRRR	TTT	333	222
SSSSSSSSSS	000	RRRRRRRRRRRR	TTT	333	222
SSSSSSSSSS	000	RRRRRRRRRRRR	TTT	333	222
SSS	000	RRR	TTT	333	222
SSS	000	RRR	TTT	333	222
SSS	000	RRR	TTT	333	222
SSS	000	RRR	TTT	333	222
SSS	000	RRR	TTT	333	222
SSS	000	RRR	TTT	333	222
SSSSSSSSSSSS	000000000	RRR	TTT	3333333333	22222222222222
SSSSSSSSSSSS	000000000	RRR	TTT	3333333333	22222222222222
SSSSSSSSSSSS	000000000	RRR	TTT	3333333333	22222222222222

```
SSSSSSSS 000000 RRRRRRRR EEEEEEEEE NN NN TTTTTTTTT RRRRRRRR YY YY
SSSSSSSS 000000 RRRRRRRR EEEEEEEEE NN NN TTTTTTTTT RRRRRRRR YY YY
SS SS 00 00 RR RR RR EE EE NN NN TT TT RR RR YY YY
SS SS 00 00 RR RR RR EE EE NN NN TT TT RR RR YY YY
SS SSSSSS 00 00 RRRRRRRR EEEEEEEEE NN NN TT TT RRRRRRRR YY YY
SS SSSSSS 00 00 RRRRRRRR EEEEEEEEE NN NN TT TT RRRRRRRR YY YY
SS SS 00 00 RR RR RR EE EE NN NN TT TT RR RR YY YY
SS SS 00 00 RR RR RR EE EE NN NN TT TT RR RR YY YY
SSSSSSSS 000000 RRR RR RR EEEEEEEEE NN NN TT TT RR RR YY YY
SSSSSSSS 000000 RRR RR RR EEEEEEEEE NN NN TT TT RR RR YY YY
.....
```

```
LL LL I I I I I SSSSSSSS
LL LL I I I I I SSSSSSSS
LL LL I I I I I
LL LL I I I I I
LL LL I I I I I
LL LL I I I I I
LL LL I I I I I
LL LL I I I I I
LL LL I I I I I
LL LL I I I I I
LL LL I I I I I
LLLLLLLLLL I I I I I SSSSSSSS
LLLLLLLLLL I I I I I SSSSSSSS
```

```
1 0001 0 MODULE SOR$ENTRY(MAIN=SOR$ENTRY,
2 0002 0 IDENT = 'V04-000',
3 0003 0 ) =
4 0004 1 BEGIN
5 0005 1
6 0006 1 *****
7 0007 1 *
8 0008 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
9 0009 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
10 0010 1 * ALL RIGHTS RESERVED.
11 0011 1 *
12 0012 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
13 0013 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
14 0014 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
15 0015 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
16 0016 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
17 0017 1 * TRANSFERRED.
18 0018 1 *
19 0019 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
20 0020 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
21 0021 1 * CORPORATION.
22 0022 1 *
23 0023 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
24 0024 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
25 0025 1 *
26 0026 1 *
27 0027 1 *****
28 0028 1
29 0029 1 ++
30 0030 1
31 0031 1 FACILITY: VAX SORT/MERGE
32 0032 1
33 0033 1 ABSTRACT:
34 0034 1
35 0035 1 This module contains the main entry to the sort/merge utility.
36 0036 1
37 0037 1 ENVIRONMENT: VAX/VMS user mode
38 0038 1
39 0039 1 AUTHOR: Peter D Gilbert, CREATION DATE: 07-Jan-1982
40 0040 1
41 0041 1 MODIFIED BY:
42 0042 1
43 0043 1
44 0044 1 T03-015 Original
45 0045 1 T03-018 Change stat names to SOR$K_xxx. PDG 4-Jan-1983
46 0046 1 T03-019 Return with the worst severity we've seen. PDG 14-Jan-1983
47 0047 1 T03-020 Change "work file size used" to "work file allocation".
48 0048 1 PDG 27-Jan-1983
49 0049 1 T03-021 Changed the name of SOR$STAT. PDG 3-Mar-1983
50 0050 1 T03-022 Reformat statistics, removing MBC and MBF. PDG 8-Jul-1983
51 0051 1 T03-023 Remove "-11" from statistics. PDG 10-Nov-1983
52 0052 1 --
```



```
54 0053 1 LIBRARY 'SYSS$LIBRARY:STARLET';
55 0054 1 LIBRARY 'SYSS$LIBRARY:XPORT';
56 0055 1
57 0056 1 %IF %DECLARED(%QUOTE $DESCRIPTOR) %THEN UNDECLARE %QUOTE $DESCRIPTOR; %FI
58 0057 1
59 0058 1 LINKAGE
60 0059 1     JSB_ONE_STAT = JSB (REGISTER=1): NOTUSED(2,3,4,5,6,7,8,9,10,11);
61 0060 1
62 0061 1 FORWARD ROUTINE
63 0062 1     COND HAND,           ! Handle exception conditions
64 0063 1     SOR_ERROR,         ! Issue an error diagnostic
65 0064 1     SOREENTRY,        ! Main entry point
66 0065 1     INIT_STATS,       ! Get initial statistics
67 0066 1     ONE_STAT: JSB_ONE_STAT, ! Get one statistic
68 0067 1     PRINT_STATS;      ! Print sort/merge statistics
69 0068 1
70 0069 1 EXTERNAL ROUTINE
71 0070 1     SOR$$COMMAND,           ! Parse command line
72 0071 1     SOR$$OUTPUT,          ! Output text
73 0072 1     SOR$SORT_MERGE:      ADDRESSING_MODE(GENERAL), ! Sort the stuff
74 0073 1     SOR$END_SORT:        ADDRESSING_MODE(GENERAL), ! Terminate sort/merge
75 0074 1     SOR$STAT:            ADDRESSING_MODE(GENERAL), ! Get a statistic
76 0075 1     LIB$FIXUP_FLT:       ADDRESSING_MODE(GENERAL),
77 0076 1     LIB$FIXUP_DEC:       ADDRESSING_MODE(GENERAL),
78 0077 1     LIB$SIGNAC:         ADDRESSING_MODE(GENERAL);
79 0078 1
80 0079 1 MACRO
81 0080 1     BASE_ = 0, 0, 0, 0 %;
82 0081 1
83 0082 1 EXTERNAL LITERAL
84 0083 1     SORT$_FACILITY;
85 0084 1 BIND
86 0085 1     SOR$_SHR_SYSERROR = SHR$_SYSERROR + STS$_SEVERE + SORT$_FACILITY ^ 16;
87 0086 1
88 0087 1 ! FAO string used to output statistics via SYSS$PUTMSG.
89 0088 1
90 0089 1 ! The following text interacts closely with the code in PRINT_STATS.
91 0090 1 ! The text can, however, be changed (translated) independent of the code, if
92 0091 1 ! the control string still uses the same FAO parameters, and text expands to
93 0092 1 ! no more than 1024 characters (a restriction of the way that the text is
94 0093 1 ! output), and lines are separated by carriage-return/line-feed pairs.
95 0094 1
96 0095 1 ! Note that the use of tab character in the text is avoided, since
97 0096 1 ! some terminals may not have tab stops at multiples of eight.
98 0097 1
99 0098 1 MACRO
100 L 0099 1     STR_STATS = %EXPAND %STRING(
101 L 0100 1         %IF %SWITCHES(DEBUG)
102 L 0101 1         %THEN '!!/!!18* VAX Sort/Merge !AC Statistics'
103 L 0102 1         %ELSE '!!/!!18* VAX Sort/Merge !+Statistics' %FI,
104 L 0103 1         '!!/'
105 L 0104 1         '!!/Records read:!!12UL',           '!!10* Input record length:!!9UL',
106 L 0105 1         '!!/Records sorted:!!10UL',        '!!10* Internal length:!!13UL',
107 L 0106 1         '!!/Records output:!!10UL',        '!!10* Output record length:!!8UL',
108 L 0107 1         '!!/Working set extent:!!6UL',      '!!10* Sort tree size:!!14UL',
109 L 0108 1         '!!/Virtual memory:!!10UL',         '!!10* Number of initial runs:!!6UL',
110 L 0109 1         '!!/Direct I/O:!!14UL',             '!!10* Maximum merge order:!!9UL',
```

SORENTRY  
V04-000

: 111 L 0110 1  
: 112 L 0111 1  
: 113 L 0112 1  
: 114 0113 1

:/Buffered I/O: !12UL',  
:/Page faults: !13UL',  
:/Elapsed time: !14xf',  
:) %;

M 6  
16-Sep-1984 00:23:12 VAX-11 Bliss-32 V4.0-742  
14-Sep-1984 13:10:43 [SORT32.SRC]SORENTRY.B32;1

Page 3  
(2)

!10\* Number of merge passes: !6UL',  
!10\* Work file allocation: !8UL',  
!7\* Elapsed CPU: !6\* !14%T',

```
116 0114 1 ! Besides information that is stored in the context area for statistics,
117 0115 1 ! a save area is used to store initial values of some statistics.
118 0116 1
119 0117 1 FIELD
120 0118 1     STAT_FIELDS =
121 0119 1         SET
122 0120 1             STAT_BUFIO = [$INTEGER], ! Buffered I/O count
123 0121 1             STAT_CPUTIM = [$INTEGER], ! CPU time
124 0122 1             STAT_START = [$SUB_BLOCK(2)], ! Start time (quadword)
125 0123 1             STAT_DIRIO = [$INTEGER], ! Direct I/O count
126 0124 1             STAT_PAGEFLTS = [$INTEGER], ! Page faults
127 0125 1             STAT_FREPOVA = [$INTEGER] ! Free page in P0 space
128 0126 1         TES;
129 0127 1 LITERAL
130 0128 1     STAT_K_SIZE = $FIELD_SET_SIZE; ! Size of save area for statistics
131 0129 1 MACRO
132 0130 1     STAT_BLOCK = BLOCK[STAT_K_SIZE] FIELD(STAT_FIELDS) %;
133 0131 1
134 0132 1 OWN
135 0133 1     CONTEXT: LONG, ! Context parameter
136 0134 1     STATS: STAT_BLOCK, ! Block to save statistics
137 0135 1     BUFIO,
138 0136 1     CPUTIM: VECTOR[2],
139 0137 1     DIRIO,
140 0138 1     PAGEFLTS,
141 0139 1     FREPOVA,
142 0140 1     WSEXTENT,
143 0141 1     SOR_SEV,
144 0142 1     SOR_STS;
145 0143 1 BIND
146 0144 1     ITMLST = UPLIT(
147 0145 1         WORD(4,JPI$-BUFIO), BUFIO, 0,
148 0146 1         WORD(4,JPI$-CPUTIM), CPUTIM, 0,
149 0147 1         WORD(4,JPI$-DIRIO), DIRIO, 0,
150 0148 1         WORD(4,JPI$-PAGEFLTS), PAGEFLTS, 0,
151 0149 1         WORD(4,JPI$-FREPOVA), FREPOVA, 0,
152 0150 1         WORD(4,JPI$-WSEXTENT), WSEXTENT, 0,
153 0151 1         0);
```



```
155 0152 1 ROUTINE COND_HAND
156 0153 1 (
157 0154 1     SIGVEC: REF BLOCK[,BYTE],      ! Signal vector
158 0155 1     MCHVEC: REF BLOCK[,BYTE]    ! Mechanism vector
159 0156 1 ) =
160 0157 1 ++
161 0158 1
162 0159 1 FUNCTIONAL DESCRIPTION:
163 0160 1
164 0161 1     Condition handler for errors occurring during sort/merge.
165 0162 1     The returned R0 is set to the error message.
166 0163 1
167 0164 1 FORMAL PARAMETERS:
168 0165 1
169 0166 1     SIGVEC.ra.r      The signal vector
170 0167 1     MCHVEC.ra.r    The mechanism vector
171 0168 1
172 0169 1 IMPLICIT INPUTS:
173 0170 1
174 0171 1     NONE
175 0172 1
176 0173 1 IMPLICIT OUTPUTS:
177 0174 1
178 0175 1     NONE
179 0176 1
180 0177 1 ROUTINE VALUE:
181 0178 1
182 0179 1     Status code.
183 0180 1
184 0181 1 SIDE EFFECTS:
185 0182 1
186 0183 1     NONE
187 0184 1
188 0185 1 --
189 0186 1 BEGIN
190 0187 1
191 0188 1     ! If we are unwinding, just return
192 0189 1
193 0190 1     IF .SIGVEC[CHFSL_SIG_NAME] EQL SS$_UNWIND THEN RETURN SS$_RESIGNAL;
194 0191 1
195 0192 1
196 0193 1     ! If SS$_ROPRAND, then try using LIB$FIXUP_FLT/DEC
197 0194 1
198 0195 1     IF .SIGVEC[CHFSL_SIG_NAME] EQL SS$_ROPRAND
199 0196 1     THEN
200 0197 1         BEGIN
201 0198 1             BUILTIN
202 0199 1                 AP,
203 0200 1                 CALLG;
204 0201 1             LOCAL
205 0202 1                 SIG_PC: REF VECTOR[,BYTE],      ! PC of bad instruction
206 0203 1                 STATUS;
207 0204 1
208 0205 1             SIG_PC = .VECTOR[SIGVEC[BASE_], .SIGVEC[CHFSL_SIG_ARGS]-1];
209 0206 1
210 0207 1             ! Repair the operand, based on the opcode
211 0208 1
```

```
212 0209 4 STATUS = (SELECTONE .SIG_PC[0] OF
213 0210 4 SET
214 0211 4 [OPS_CVTTP, OPS_CVTSP]: CALLG(.AP, LIB$FIXUP_DEC);
215 0212 4 [OPS_CMPF, OPS_CMPD, OPS_ESCD]: CALLG(.AP, LIB$FIXUP_FLT);
216 0213 4 [OTHERWISE]: 0;
217 0214 4 TES);
218 0215 4
219 0216 4 IF .STATUS EQL SS$_NORMAL
220 0217 4 THEN
221 0218 4 BEGIN
222 0219 4
223 0220 4 We managed to repair the problem.
224 0221 4 However, we should let the user know that an error occurred.
225 0222 4
226 0223 4 EXTERNAL LITERAL SOR$_ROPRAND;
227 0224 4 LIB$SIGNAL(SOR$_ROPRAND);
228 0225 4 RETURN SS$_NORMAL;
229 0226 4 END;
230 0227 4 END;
231 0228 4
232 0229 4 ! Set the returned R0 value
233 0230 4 !
234 0231 4 MCHVEC[CHFS$_MCH_SAVR0] = .SIGVEC[CHFS$_SIG_NAME];
235 0232 4
236 0233 4 ! Hang onto the worst error we've seen
237 0234 4 !
238 0235 4 BEGIN
239 0236 4 BIND CVT_SEV = UPLIT BYTE(2,0,3,1,4,5,6,7): VECTOR[,BYTE];
240 0237 4 LOCAL SEV;
241 0238 4 SEV = .CVT_SEV[BLOCK[SIGVEC[CHFS$_SIG_NAME],STSV_SEVERITY;[,BYTE]]];
242 0239 4 IF .SEV GTRU .SOR_SEV
243 0240 4 THEN
244 0241 4 BEGIN
245 0242 4 SOR_SEV = .SEV;
246 0243 4 SOR_STS = .SIGVEC[CHFS$_SIG_NAME] OR STSM_INHIB_MSG;
247 0244 4 END;
248 0245 4
249 0246 4 END;
250 0247 4
251 0248 4 ! Resignal the error. If the severity of the error is Success, Info,
252 0249 4 ! Warning, or Error, execution will continue.
253 0250 4 !
254 0251 4 RETURN SS$_RESIGNAL;
255 0252 4
256 0253 4 END;
257 0254 4
```

```
.TITLE SORENTY
.IDENT \V04-000\
.PSECT $PLIT$,NOWRT,NOEXE,2
```

```
040C 0004 00000 P.AAA: .WORD 4, 1036
00000000 00004 .ADDRESS BUFIO
00000000 00008 .LONG 0
0407 0004 0000C .WORD 4, 1031
```

:  
:  
:  
:  
:



```
00000000' 00010 .ADDRESS CPUTIM
00000000 00014 .LONG 0
040B 0004 00018 .WORD 4, 1035
00000000' 0001C .ADDRESS DIRIO
00000000 00020 .LONG 0
040A 0004 00024 .WORD 4, 1034
00000000' 00028 .ADDRESS PAGEFLTS
00000000 0002C .LONG 0
0404 0004 00030 .WORD 4, 1028
00000000' 00034 .ADDRESS FREPOVA
00000000 00038 .LONG 0
0416 0004 0003C .WORD 4, 1046
00000000' 00040 .ADDRESS WSEXTENT
00000000 00044 .LONG 0, 0
07 06 05 00000000 00000000 0004C P.AAB: .BYTE 2, 0, 3, 1, 4, 5, 6, 7
```

.PSECT \$OWNS\$,NOEXE,2

```
00000 CONTEXT: .BLKB 4
00004 STATS: .BLKB 32
00024 BUFIO: .BLKB 4
00028 CPUTIM: .BLKB 8
00030 DIRIO: .BLKB 4
00034 PAGEFLTS:
          .BLKB 4
00038 FREPOVA: .BLKB 4
0003C WSEXTENT:
          .BLKB 4
00040 SOR_SEV: .BLKB 4
00044 SOR_STS: .BLKB 4
```

ITMLST=  
CVT\_SEV=

P.AAA  
P.AAB

```
.EXTRN SOR$$COMMAND, SOR$$OUTPUT
.EXTRN SOR$$SORT_MERGE, SOR$END_SORT
.EXTRN SOR$$STAT, LIB$FIXUP_FLT
.EXTRN LIB$FIXUP_DEC, LIB$SIGNAL
.EXTRN SORT$_FACILITY, SOR$_ROPRAND
```

.PSECT \$CODE\$,NOWRT,2

```
0000 00000 COND_HAND:
00000920 51 04 AC D0 00002 .WORD Save nothing
8F 04 A1 D1 00006 MOVL SIGVEC, R1
00000454 8F 04 76 13 0000E CMPL 4(R1), #2336
04 A1 D1 00010 BEQL 7$
4E 12 00018 CMPL 4(R1), #1108
61 D0 0001A BNEQ 6$
50 61 0001D MOVL (R1), R0
51 FC A140 D0 0001D MOVL -4(R1)[R0], SIG_PC
09 61 91 00022 CMPB (SIG_PC), #9
05 13 00025 BEQL 1$
26 61 91 00027 CMPB (SIG_PC), #38
09 12 0002A BNEQ 2$
00000000G 00 6C FA 0002C 1$: CALLG (AP), LIB$FIXUP_DEC
1D 11 00033 BRB 5$
51 8F 61 91 00035 2$: CMPB (SIG_PC), #81
```

71	8F	0C	13	00039	BEQL	3\$		
		61	91	0003B	CMPB	(SIG_PC), #113		
FD	8F	06	13	0003F	BEQL	3\$		
		61	91	00041	CMPB	(SIG_PC), #253		
00000000G	00	09	12	00045	BNEQ	4\$		
		6C	FA	00047	CALLG	(AP), LIB\$FIXUP_FLT		
		02	11	0004E	BRB	5\$		
	01	50	D4	00050	CLRL	STATUS		0213
		50	D1	00052	CMPL	STATUS, #1		0216
		11	12	00055	BNEQ	6\$		
00000000G	00	8F	DD	00057	PUSHL	#SOR\$ ROPRAND		0224
	50	01	FB	0005D	CALLS	#1, LIB\$SIGNAL		
		01	D0	00064	MOVL	#1, R0		0225
			04	00067	RET			
	50	08	AC	D0 00068	MOVL	MCHVEC, R0		0232
	51	04	AC	D0 0006C	MOVL	SIGVEC, R1		
50	04	04	A1	D0 00070	MOVL	4(R1), 12(R0)		
	03	00	EF	00075	EXTZV	#0, #3, 4(R1), R0		0239
	50	0000' CF	40	9A 0007B	MOVZBL	CVT_SEV[R0], SEV		
	0000'	CF	50	D1 00081	CMPL	SEV, SOR_SEV		0240
			10	1B 00086	BLEQU	8\$		
	0000'	CF	50	D0 00088	MOVL	SEV, SOR_SEV		0243
0000' CF	04	A1	10000000	8F C9 0008D	BISL3	#268435456, 4(R1), SOR_STS		0244
		50	0918	8F 3C 00098	MOVZWL	#2328, R0		0252
			04	0009D	RET			0254

; Routine Size: 158 bytes, Routine Base: \$CODE\$ + 0000

```

: 259 0255 1 ROUTINE SOR_ERROR(ERR) =
: 260 0256 1
: 261 0257 1 ++
: 262 0258 1
: 263 0259 1 FUNCTIONAL DESCRIPTION:
: 264 0260 1
: 265 0261 1 This routine signals an error diagnostic.
: 266 0262 1
: 267 0263 1 FORMAL PARAMETERS:
: 268 0264 1
: 269 0265 1 Parameters passed to LIB$SIGNAL.
: 270 0266 1
: 271 0267 1 IMPLICIT INPUTS:
: 272 0268 1
: 273 0269 1 NONE
: 274 0270 1
: 275 0271 1 IMPLICIT OUTPUTS:
: 276 0272 1
: 277 0273 1 NONE
: 278 0274 1
: 279 0275 1 ROUTINE VALUE:
: 280 0276 1
: 281 0277 1 System status (first parameter of signalled status), with the
: 282 0278 1 INHIB_MSG bit set.
: 283 0279 1
: 284 0280 1 SIDE EFFECTS:
: 285 0281 1
: 286 0282 1 The image may be exited due to the error.
: 287 0283 1
: 288 0284 1 --
: 289 0285 2 BEGIN
: 290 0286 2 BUILTIN
: 291 0287 2 AP,
: 292 0288 2 CALLG;
: 293 0289 2 LOCAL
: 294 0290 2 STATUS;
: 295 0291 2 CALLG(.AP, LIB$SIGNAL);
: 296 0292 2 RETURN .ERR OR ST$SM_INHIB_MSG;
: 297 0293 1 END;

```

```

0000 00000 SOR_ERROR:
50 00000000G 00 6C FA 00002 .WORD Save nothing
04 AC 10000000 8F C9 00009 CALLG (AP), LIB$SIGNAL
04 00012 B1SL3 #268435456, ERR, R0
RET

```

: Routine Size: 19 bytes, Routine Base: \$CODE\$ + 009E

: 0255  
: 0291  
: 0292  
: 0293



```

0299 1 GLOBAL ROUTINE SOREENTRY =
0300 1 ++
0301 1
0302 1 FUNCTIONAL DESCRIPTION:
0303 1
0304 1     This is the main entry point to the SORT/MERGE utilities.
0305 1     This routine does the following:
0306 1
0307 1     Parse the command line.
0308 1     Process the specification file.
0309 1     Use the callable sort/merge routines to finish processing.
0310 1     Print statistics, if requested.
0311 1     Release allocated resources.
0312 1
0313 1 FORMAL PARAMETERS:
0314 1
0315 1     NONE
0316 1
0317 1 IMPLICIT INPUTS:
0318 1
0319 1     NONE
0320 1
0321 1 IMPLICIT OUTPUTS:
0322 1
0323 1     NONE
0324 1
0325 1 ROUTINE VALUE:
0326 1
0327 1     System status code.
0328 1
0329 1 SIDE EFFECTS:
0330 1
0331 1     NONE
0332 1
0333 1 --
0334 1 BEGIN
0335 1 LOCAL
0336 1     STATISTICS,           ! Flag for whether statistics requested
0337 1     SORT_FLAG,           ! Flag indicating sort (not merge)
0338 1     STATUS;              ! Status
0339 1
0340 1     ! Initialize the severity and message to success
0341 1     SOR_SEV = 0;
0342 1     SOR_STS = $$$_NORMAL;
0343 1
0344 1     ! Establish a condition handler
0345 1     (BUILTIN FP; .FP = COND_HAND);
0346 1
0347 1     ! Clear the context longword
0348 1     CONTEXT = 0;
0349 1
0350 1
0351 1
0352 1
0353 1
0354 1
0355 1

```

```

: Initialize the statistics
STATUS = INIT_STATS();
IF NOT .STATUS THEN RETURN .STATUS;

: Call SOR$$COMMAND to process the command line, call SPEC_FILE,
: call PASS_FILES, and call INIT_SORT or INIT_MERGE.
: The context parameter is not referenced by SOR$$COMMAND, it is
: just passed to the callable interface routines.
: SOR$$COMMAND sets or clears SORT_FLAG depending on whether
: we were invoked for a sort or a merge, respectively.
: SOR$$COMMAND sets or clears STATISTICS depending on whether
: statistics were requested for the sort/merge.
STATUS = SOR$$COMMAND(
    CONTEXT,
    SORT_FLAG,
    STATISTICS,
    (BUILTIN AP, AP));
IF NOT .STATUS THEN RETURN .STATUS;

IF .SORT_FLAG
THEN
    BEGIN
        : Call SORT_MERGE
        STATUS = SOR$SORT_MERGE(CONTEXT);
        IF NOT .STATUS THEN RETURN .STATUS;
    END;

: Put out the statistics, if requested.
IF .STATISTICS
THEN
    BEGIN
        STATUS = PRINT_STATS();
        IF NOT .STATUS THEN RETURN .STATUS;
    END;

: Call END_SORT to clean up after ourselves
STATUS = SOR$END_SORT(CONTEXT);
IF NOT .STATUS THEN RETURN .STATUS;

: Return the worst error we've seen
RETURN .SOR_STS;
END;
```

			0004	00000	.ENTRY	SORSEENTRY, Save R2	0294
	52	0000'	CF	9E 00002	MOVAB	CONTEXT, R2	
	5E		08	C2 00007	SUBL2	#8, SP	
		40	A2	D4 0000A	CLRL	SOR_SEV	0340
44	A2		01	D0 0000D	MOVL	#1, SOR_STS	0341
	6D	FF3A	CF	9E 00011	MOVAB	COND_HARD, (FP)	0345
			62	D4 00016	CLRL	CONTEXT	0349
0000V	CF		00	FB 00018	CALLS	#0, INIT_STATS	0353
	3D		50	E9 0001D	BLBC	STATUS, 3\$	0354
			5C	DD 00020	PUSHL	AP	0373
		04	AE	9F 00022	PUSHAB	STATISTICS	0369
		0C	AE	9F 00025	PUSHAB	SORT_FLAG	
			52	DD 00028	PUSHL	R2	
0000G	CF		04	FB 0002A	CALLS	#4, SOR\$\$COMMAND	
	2B		50	E9 0002F	BLBC	STATUS, 3\$	0374
	0C	04	AE	E9 00032	BLBC	SORT_FLAG, 1\$	0377
			52	DD 00036	PUSHL	R2	0383
00000000G	00		01	FB 00038	CALLS	#1, SOR\$SORT_MERGE	
	1B		50	E9 0003F	BLBC	STATUS, 3\$	0384
	08		6E	E9 00042 1\$:	BLBC	STATISTICS, 2\$	0390
0000V	CF		00	FB 00045	CALLS	#0, PRINT_STATS	0393
	10		50	E9 0004A	BLBC	STATUS, 3\$	0394
			52	DD 0004D 2\$:	PUSHL	R2	0400
00000000G	00		01	FB 0004F	CALLS	#1, SOR\$END_SORT	
	04		50	E9 00056	BLBC	STATUS, 3\$	0401
	50	44	A2	D0 00059	MOVL	SOR_STS, R0	0406
			04	0005D 3\$:	RET		0407

; Routine Size: 94 bytes, Routine Base: \$CODE\$ + 00B1



```
414 0408 1 ROUTINE INIT_STATS =
415 0409 1
416 0410 1 **
417 0411 1
418 0412 1 FUNCTIONAL DESCRIPTION:
419 0413 1
420 0414 1     This routine initializes sort/merge statistics.
421 0415 1
422 0416 1 FORMAL PARAMETERS:
423 0417 1
424 0418 1     NONE
425 0419 1
426 0420 1 IMPLICIT INPUTS:
427 0421 1
428 0422 1     NONE
429 0423 1
430 0424 1 IMPLICIT OUTPUTS:
431 0425 1
432 0426 1     NONE
433 0427 1
434 0428 1 ROUTINE VALUE:
435 0429 1
436 0430 1     System status value
437 0431 1
438 0432 1 SIDE EFFECTS:
439 0433 1
440 0434 1     NONE
441 0435 1
442 0436 1 --
443 0437 2 BEGIN
444 0438 2 LOCAL
445 0439 2     STATUS;
446 0440 2
447 0441 2     ! Get the statistics
448 0442 2
449 0443 2     STATUS = $GETJPI(ITMLST=ITMLST);
450 0444 2     IF NOT .STATUS THEN RETURN SOR_ERROR(SOR$ SHR_SYSEERROR, 0, .STATUS);
451 0445 2     STATUS = $GETTIM(TIMADR=STATS[STAT_START]);
452 0446 2     IF NOT .STATUS THEN RETURN SOR_ERROR(SOR$ SHR_SYSEERROR, 0, .STATUS);
453 0447 2
454 0448 2     STATS[STAT_BUFIO] = .BUFIO;
455 0449 2     STATS[STAT_CPUTIM] = .CPUTIM;
456 0450 2     STATS[STAT_DIRIO] = .DIRIO;
457 0451 2     STATS[STAT_PAGEFLTS] = .PAGEFLTS;
458 0452 2     STATS[STAT_FREPOVA] = .FREPOVA;
459 0453 2
460 0454 2 RETURN SSS_NORMAL;
461 0455 1 END;
```

.EXTRN SYS\$GETJPI, SYS\$GETTIM

000C 00000 INIT\_STATS:

53

0000'

CF 9E 00002  
7E 7C 00007.WORD  
MOVAB  
CLRQSave R2,R3  
STATS+12, R3  
-(SP)

: 0408

: 0443

				0000'	7E D4 00009	CLRL	-(SP)	
					CF 9F 0000B	PUSHAB	ITMLST	
					7E 7C 0000F	CLRL	-(SP)	
					7E D4 00011	CLRL	-(SP)	
	00000000G	00			07 FB 00013	CALLS	#7, SYSSGETJPI	
		52			50 D0 0001A	MOVL	R0, STATUS	
		0F			52 E9 0001D	BLBC	STATUS, 1\$	0444
					53 DD 00020	PUSHL	R3	0445
	00000000G	00			01 FB 00022	CALLS	#1, SYSSGETTIM	
		52			50 D0 00029	MOVL	R0, STATUS	
		10			52 E8 0002C	BLBS	STATUS, 2\$	0446
					52 DD 0002F	PUSHL	STATUS	
					7E D4 00031	CLRL	-(SP)	
					8F DD 00033	PUSHL	#<<SORT\$ FACILITY@16>+4532>	
	FF51	CF		00000000*	03 FB 00039	CALLS	#3, SOR_ERROR	
					04 0003E	RET		
F5	A3	20		02	A3 F0 0003F	INSV	BUF10, #2, #32, STATS+1	0448
F9	A3	20		02	A3 F0 00046	INSV	CPUTIM, #2, #32, STATS+5	0449
				08	A3 7D 0004D	MOVQ	DIR10, STATS+20	0450
				10	A3 D0 00052	MOVL	FREPOVA, STATS+28	0452
				50	01 D0 00057	MOVL	#1, R0	0454
					04 0005A	RET		0455

: Routine Size: 91 bytes, Routine Base: \$CODE\$ + 010F

```

463 0456 1 ROUTINE ONE_STAT
464 0457 1 (
465 0458 1 CODE
466 0459 1 ): JSB_ONE_STAT =
467 0460 1 ++
468 0461 1
469 0462 1 FUNCTIONAL DESCRIPTION:
470 0463 1
471 0464 1 This routine gets one sort/merge statistic.
472 0465 1
473 0466 1 FORMAL PARAMETERS:
474 0467 1
475 0468 1 CODE.rl.v Code of statistic to get
476 0469 1
477 0470 1 IMPLICIT INPUTS:
478 0471 1
479 0472 1 NONE
480 0473 1
481 0474 1 IMPLICIT OUTPUTS:
482 0475 1
483 0476 1 NONE
484 0477 1
485 0478 1 ROUTINE VALUE:
486 0479 1
487 0480 1 Value of the statistic
488 0481 1
489 0482 1 SIDE EFFECTS:
490 0483 1
491 0484 1 NONE
492 0485 1
493 0486 1 --
494 0487 2 BEGIN
495 0488 2 LOCAL
496 0489 2 RESULT,
497 0490 2 STATUS;
498 0491 2
499 0492 2 RESULT = 0;
500 0493 2 STATUS = SOR$STAT(CODE, RESULT, CONTEXT);
501 0494 2
502 0495 2 IF NOT .STATUS THEN SOR_ERROR(.STATUS);
503 0496 2
504 0497 2 RETURN .RESULT;
505 0498 1 END;

```

```

51 DD 00000 ONE_STAT:
0000' 7E D4 00002 PUSHL R1
04 CF 9F 00004 CLRL RESULT
0C AE 9F 00008 PUSHAB CONTEXT
0C AE 9F 0000B PUSHAB RESULT
00000000G 00 03 FB 0000E PUSHAB CODE
07 50 EB 00015 CALLS #3, SOR$STAT
50 DD 00018 BLBS STATUS, 1$
PUSHL STATUS

```

0456  
0492  
0493  
  
0495



SORENTRY  
V04-000

M 7  
16-Sep-1984 00:23:12 VAX-11 B11ss-32 V4.0-742  
14-Sep-1984 13:10:43 [SORT32.SRC]SORENTRY.B32;1

Page 16  
(8)

FF15 CF  
SO  
SE

01 FB 0001A  
8E DO 0001F 18:  
04 CO 00022  
05 00025

CALLS #1, SOR\_ERROR  
MOVL RESULT, -R0  
ADDL2 #4, SP  
RSB

: 0497  
: 0498  
:

; Routine Size: 38 bytes, Routine Base: \$CODES + 016A

```
0499 1 GLOBAL ROUTINE PRINT_STATS =
0500 1
0501 1 ++
0502 1
0503 1 FUNCTIONAL DESCRIPTION:
0504 1
0505 1     This routine prints sort/merge statistics.
0506 1
0507 1 FORMAL PARAMETERS:
0508 1
0509 1     NONE
0510 1
0511 1 IMPLICIT INPUTS:
0512 1
0513 1     NONE
0514 1
0515 1 IMPLICIT OUTPUTS:
0516 1
0517 1     NONE
0518 1
0519 1 ROUTINE VALUE:
0520 1
0521 1     System status value
0522 1
0523 1 SIDE EFFECTS:
0524 1
0525 1     NONE
0526 1
0527 1 --
0528 1 BEGIN
0529 1     %IF NOT %DECLARED(COM_K_BPERPAGE)
0530 1         %THEN LITERAL COM_K_BPERPAGE = 512; %FI
0531 1 BUILTIN
0532 1     EMUL;
0533 1 LOCAL
0534 1     FINIS: VECTOR[2],
0535 1     CTRSTR: VECTOR[2],
0536 1     STATUS;
0537 1 MACRO
0538 1     S_(X) =
0539 1         (EXTERNAL LITERAL %NAME('SOR$K_',X): UNSIGNED(5);
0540 1         ONE_STAT(%NAME('SOR$K_',X))) %;
0541 1
0542 1     ! Get the statistics
0543 1     !
0544 1     STATUS = $GETJPI(ITMLST=ITMLST);
0545 1     IF NOT .STATUS THEN RETURN .STATUS;
0546 1     STATUS = $GETTIM(TIMADR=FINIS[0]);
0547 1     IF NOT .STATUS THEN RETURN .STATUS;
0548 1
0549 1     ! Do a quadword subtract to compute the elapsed time.
0550 1     !
0551 1     BEGIN
0552 1     BIND
0553 1         T = STATS[STAT_START]: VECTOR[2];
0554 1         IF .FINIS[0] LSSU T[0] THEN FINIS[1] = .FINIS[1] - 1;
```

```
FINIS[0] = .FINIS[0] - .T[0];
FINIS[1] = .FINIS[1] - .T[1];
END;

! Compute the elapsed CPU time, and convert it from 10-millisecond units to
! 100-nanosecond units (the standard VMS date/time format) by multiplying
! by 100000.
CPUTIME[0] = .CPUTIME[0] - .STATS[STAT_CPUTIME];
EMUL(CPUTIME[0], %REF(100000), %REF(0), CPUTIME[0]);

! Format and output the statistics
CTRSTR[0] = %CHARCOUNT(STR_STATS);
CTRSTR[1] = UPLIT BYTE(STR_STATS);
STATUS = SOR$OUTPUT(CTRSTR,
    S_(IDENT),           ! Address of ASCII ident string
    S_(REC_INP),          ! Records input
    S_(LRL_INP),          ! Record length
    S_(REC_SOR),          ! Records sorted
    S_(LRL_INT),          ! Internal record length
    S_(REC_OUT),          ! Records output
    S_(LRL_OUT),          ! Output record length
    %SEXTENT,             ! Working-set
    S_(NODES),            ! Nodes in tree
    (%FREPOVA - .STATS[STAT_FREPOVA]) / COM_K_BPERPAGE, ! Memory used
    S_(INI_RUNS),          ! Number of runs
    S_(DIRIO - .STATS[STAT_DIRIO]), ! Direct I/O
    S_(MRG_ORDER),         ! Merge order
    S_(BUFIO - .STATS[STAT_BUFIO]), ! Buffered I/O
    S_(MRG_PASSES),        ! Merge passes
    S_(PAGEFLT - .STATS[STAT_PAGEFLT]), ! Page faults
    S_(WRK_ALQ),           ! Work file allocation
    FINIS[0],              ! Wall time
    CPUTIME[0],            ! CPU time
    0);                    ! Dummy

IF NOT .STATUS THEN RETURN SOR_ERROR(SOR$_SHR_SYSERROR, 0, .STATUS);
RETURN $$$_NORMAL;
END;
```

```
74 72 6F 53 20 58 41 56 20 2A 38 31 21 2F 21 00054 P.AAC: .ASCII \!/:18* VAX Sort/Merge !+Statistics!/:Re\
73 69 74 61 74 53 2B 21 20 65 67 72 65 4D 2F 00063
    65 52 2F 21 2F 21 73 63 69 74 00072
55 32 31 21 3A 64 61 65 72 20 73 64 72 6F 63 0007C .ASCII \cords read:!12UL!10* Input record length\
63 65 72 20 74 75 70 6E 49 20 2A 30 31 21 4C 0008B
    68 74 67 6E 65 6C 20 64 72 6F 0009A
20 73 64 72 6F 63 65 52 2F 21 4C 55 39 21 3A 000A4 .ASCII \!:9UL!/Records sorted:!10UL!10* Internal\
30 31 21 4C 55 30 31 21 3A 64 65 74 72 6F 73 000B3
    6C 61 6E 72 65 74 6E 49 20 2A 000C2
2F 21 4C 55 33 31 21 3A 68 74 67 6E 65 6C 20 000CC .ASCII \ length:!13UL!/Records output:!10UL!10* \
```



```
3A 74 75 70 74 75 6F 20 73 64 72 6F 63 65 52 000DB
20 6C 20 64 72 6F 63 65 72 20 74 75 70 74 75 4F 000EA
72 6F 57 2F 21 4C 55 38 21 3A 68 74 67 6E 65 000F4
20 2A 30 31 21 4C 55 74 65 73 20 67 6E 69 6B 00103
3A 65 7A 69 73 20 65 65 72 74 20 74 72 6F 53 00112
72 69 56 2F 21 4C 55 34 31 21 4C 55 34 31 21 0011C
30 31 21 3A 79 72 6F 6D 65 6D 20 6C 61 75 74 0012B
6F 20 72 65 62 6D 75 4E 20 2A 30 31 21 4C 55 0013A
65 72 69 44 2F 21 4C 55 36 21 3A 73 6E 69 20 66 00144
30 31 21 4C 55 34 31 21 3A 4F 2F 49 20 66 6E 75 72 00153
55 39 21 3A 72 65 64 72 65 66 66 75 42 2F 21 4C 00162
4F 2F 49 20 64 2A 30 31 21 4C 55 32 31 21 3A 0016C
67 72 65 6D 20 66 6F 20 72 65 62 6D 75 4E 20 65 0017B
2F 21 4C 55 36 21 3A 73 65 73 73 61 70 20 65 0018A
72 6F 57 20 2A 30 31 21 4C 55 33 31 21 3A 73 00194
69 74 61 63 6F 6C 6C 61 20 65 6C 69 66 20 6B 001A3
31 21 20 3A 65 6D 69 74 20 64 65 73 70 61 6C 001B2
20 64 65 73 70 61 6C 65 54 25 34 001BC
31 21 20 3A 65 6D 69 74 20 64 65 73 70 61 6C 001CB
20 64 65 73 70 61 6C 65 54 25 34 001DA
31 21 20 3A 65 6D 69 74 20 64 65 73 70 61 6C 001E4
20 64 65 73 70 61 6C 65 54 25 34 001F3
31 21 20 3A 65 6D 69 74 20 64 65 73 70 61 6C 00202
20 64 65 73 70 61 6C 65 54 25 34 0020C
31 21 20 3A 65 6D 69 74 20 64 65 73 70 61 6C 0021B
20 64 65 73 70 61 6C 65 54 25 34 0022A
31 21 20 3A 65 6D 69 74 20 64 65 73 70 61 6C 00234
20 64 65 73 70 61 6C 65 54 25 34 0023A
```

```
.ASCII \Output record length:!8UL!/Working set e\
.ASCII \xtent:!6UL!10* Sort tree size:!14UL!/Vir\
.ASCII \tual memory:!10UL!10* Number of initial \
.ASCII \runs:!6UL!/Direct I/O:!14UL!10* Maximum \
.ASCII \merge order:!9UL!/Buffered I/O:!12UL!10*\
.ASCII \ Number of merge passes:!6UL!/Page fault\
.ASCII \s:!13UL!10* Work file allocation:!8UL!/E\
.ASCII \lapsed time: !14%T!7* Elapsed CPU:!6* !1\
.ASCII \4%T\
```

T=

```
STATS+12
.EXTRN SORSK_IDENT, SORSK_REC_INP
.EXTRN SORSK_LRL_INP, SORSK_REC_SOR
.EXTRN SORSK_LRL_INT, SORSK_REC_OUT
.EXTRN SORSK_LRL_OUT, SORSK_NODES
.EXTRN SORSK_INI_RUNS, SORSK_MRG_ORDER
.EXTRN SORSK_MRG_PASSES
.EXTRN SORSK_WRK_ALQ
```

.PSECT \$CODE\$,NOWRT,2

```
.ENTRY PRINT_STATS, Save R2,R3,R4
MOVAB ONE_STAT, R4
MOVAB CPUTIM, R3
SUBL2 #16, SP
CLRQ -(SP)
CLRL -(SP)
PUSHAB ITMLST
CLRQ -(SP)
CLRL -(SP)
CALLS #7, SYSSGETJPI
MOVL R0, STATUS
BLBC STATUS, 1$
PUSHAB FINIS
CALLS #1, SYSSGETTIM
MOVL R0, STATUS
BLBS STATUS, 2$
MOVL STATUS, R0
RET
```

```
54 D5 001C 00000
53 0000' AF 9E 00002
5E 10 C2 0000B
7E 7C 0000E
7E D4 00010
CF 9F 00012
7E 7C 00016
7E D4 00018
07 FB 0001A
50 D0 00021
52 E9 00024
DB AE 9F 00027
01 FB 0002A
50 D0 00031
52 E8 00034
52 D0 00037
04 0003A
```

1\$:

0499

0544

0545

0546

0547

50	E1	A3	E8	A3	08	AE	D1	0003B	2\$:	CMPL	FINIS, T	0555
63						03	1E	00040		BGEQU	3\$	
			08	AE	OC	AE	D7	00042		DECL	FINIS+4	
			OC	AE	E8	A3	C2	00045	3\$:	SUBL2	T, FINIS	0556
				20	EC	A3	C2	0004A		SUBL2	T+4, FINIS+4	0557
				63		02	EE	0004F		EXTV	#2, #32, STATS+5, R0	0565
		00	000186A0	8F		50	C2	00055		SUBL2	R0, CPUTIM	
				6E		63	7A	00058		EMUL	CPUTIM, #100000, #0, CPUTIM	0566
			04	AE	01E3	8F	3C	00061		MOVZWL	#483, CTRSTR	0571
					0000	CF	9E	00066		MOVAB	P.AAC, CTRSTR+4	0572
						7E	D4	0006C		CLRL	-(SP)	0573
						53	DD	0006E		PUSHL	R3	0592
					10	AE	9F	00070		PUSHAB	FINIS	0591
				51		00G	9A	00073		MOVZBL	S^SOR\$K_WRK_ALQ, R1	0590
						64	16	00076		JSB	ONE_STAT	
		7E	OC	A3	F4	50	DD	00078		PUSHL	R0	
				51		00G	9A	0007A		SUBL3	STATS+24, PAGEFLTS, -(SP)	0589
						64	16	00083		MOVZBL	S^SOR\$K_MRG_PASSES, R1	0588
						50	DD	00085		JSB	ONE_STAT	
50	DD	A3		20		02	EE	00087		PUSHL	R0	
		7E	FC	A3		50	C3	0008D		EXTV	#2, #32, STATS+1, R0	0587
				51		00G	9A	00092		SUBL3	R0, BUFIO, -(SP)	
						64	16	00095		MOVZBL	S^SOR\$K_MRG_ORDER, R1	0586
						50	DD	00097		JSB	ONE_STAT	
		7E	08	A3	F0	AE	C3	00099		PUSHL	R0	
				51		00G	9A	0009F		SUBL3	STATS+20, DIRIO, -(SP)	0585
						64	16	000A2		MOVZBL	S^SOR\$K_INI_RUNS, R1	0584
						50	DD	000A4		JSB	ONE_STAT	
		50	10	A3	F8	AE	C3	000A6		PUSHL	R0	
		7E		50	00000200	8F	C7	000AC		SUBL3	STATS+28, FREPOVA, R0	0583
				51		00G	9A	000B4		DIVL3	#512, R0, -(SP)	
						64	16	000B7		MOVZBL	S^SOR\$K_NODES, R1	0582
						50	DD	000B9		JSB	ONE_STAT	
					14	AE	DD	000BB		PUSHL	R0	
				51		00G	9A	000BE		PUSHL	WSEXTENT	0581
						64	16	000C1		MOVZBL	S^SOR\$K_LRL_OUT, R1	0580
						50	DD	000C3		JSB	ONE_STAT	
				51		00G	9A	000C5		PUSHL	R0	
						64	16	000C8		MOVZBL	S^SOR\$K_REC_OUT, R1	0579
						50	DD	000CA		JSB	ONE_STAT	
				51		00G	9A	000CC		PUSHL	R0	
						64	16	000CF		MOVZBL	S^SOR\$K_LRL_INT, R1	0578
						50	DD	000D1		JSB	ONE_STAT	
				51		00G	9A	000D3		PUSHL	R0	
						64	16	000D6		MOVZBL	S^SOR\$K_REC_SOR, R1	0577
						50	DD	000D8		JSB	ONE_STAT	
				51		00G	9A	000DA		PUSHL	R0	
						64	16	000DD		MOVZBL	S^SOR\$K_LRL_INP, R1	0576
						50	DD	000DF		JSB	ONE_STAT	
				51		00G	9A	000E1		PUSHL	R0	
						64	16	000E4		MOVZBL	S^SOR\$K_REC_INP, R1	0575
						50	DD	000E6		JSB	ONE_STAT	
				51		00G	9A	000E8		PUSHL	R0	
						64	16	000EB		MOVZBL	S^SOR\$K_IDENT, R1	0574
						50	DD	000ED		JSB	ONE_STAT	
					50	AE	9F	000EF		PUSHL	R0	
										PUSHAB	CTRSTR	0573

```

0000G CF      15 FB 000F2    CALLS #21, SOR$$OUTPUT
      52      50 DO 000F7    MOVL  R0, STATUS
      10      52 E8 000FA    BLBS  STATUS, 4$
      00000000* 52 DD 000FD    PUSHL STATUS
      FF34 C4  8F DD 000FF    CLRL  -(SP)
      50      03 FB 00101    PUSHL #<<SORT$ FACILITY@16>+4532>
      01      04 FB 00107    CALLS #3, SOR_ERROR
      04      04 DO 0010C    RET
      04      04 DO 0010D    MOVL  #1, R0
      04      04 DO 00110    RET
  
```

0595

0596

0597

; Routine Size: 273 bytes, Routine Base: \$CODE\$ + 0190

```

: 606      0598 1
: 607      0599 1 END
: 608      0600 0 ELUDOM
  
```

# PSECT SUMMARY

Name	Bytes	Attributes
\$OWNS	72	NOVEC, WRT, RD, NOEXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)
\$PLITS	567	NOVEC, NOWRT, RD, NOEXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)
\$CODE\$	673	NOVEC, NOWRT, RD, EXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)

# Library Statistics

File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
\$255\$DUA28:[SYSLIB]STARLET.L32;1	9776	27	0	581	00:01.0
\$255\$DUA28:[SYSLIB]XPORT.L32;1	590	20	3	252	00:00.6

# COMMAND QUALIFIERS

; BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/NOTRACE/LIS=LIS\$:SORENTRY/OBJ=OBJ\$:SORENTRY MSRC\$:SORENTRY/UPDATE=(ENH\$:SORENTRY)

```

: Size:      673 code + 639 data bytes
: Run Time:   00:15.6
: Elapsed Time: 00:50.7
: Lines/CPU Min: 2312
: Lexemes/CPU-Min: 22959
  
```



SORSEENTRY  
V04-000

F 8  
16-Sep-1984 00:23:12

VAX-11 Bliss-32 V4.0-742

Page 22

; Memory Used: 127 pages  
; Compilation Complete

SO  
VO



0364 AH-BT13A-SE  
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION  
CONFIDENTIAL AND PROPRIETARY